

REMARKS/ARGUMENTS

Claims 1-28 are pending in this application. Claims 1, 8, 13, 20, 27 and 28 are amended in the present Response. Claims 1, 6-8, 12, 13, 18-20 and 24-28 are rejected under 35 U.S.C. 102 as purportedly being anticipated by Cray, Jr. (U.S. Patent No. 4,128,880). Claims 2-5, 9-11, 14-17 and 21-23 are rejected under 35 U.S.C. 103 as purportedly being unpatentable over Cray in view of Laudon et al. (Interleaving: a Multithreading Technique Targeting Multiprocessor and Workstation).

REJECTIONS UNDER 35 U.S.C. § 102

Claim 1

Applicants respectfully submit that claim 1, as amended, overcomes the anticipation rejection based on Cray. Claim 1 recites:

1. A programmable processor comprising:

a data path capable of transmitting data;

an external interface operable to receive data from an external source and communicate the received data over the data path;

a register file containing a plurality of registers each having a register width, the register file coupled to the data path and configured to support processing of a plurality of threads and to store a plurality of multiple-bit data elements in partitioned fields, each of the multiple-bit data elements having an elemental width smaller than the register width;

an execution unit coupled to the data path, the execution unit configured to execute a plurality of instruction streams from the plurality of threads, each instruction stream including a single instruction that specifies an arithmetic operation to cause multiple instances of the arithmetic operation to be performed, each instance of the arithmetic operation to be performed using a different one of the plurality of multiple-bit data elements in partitioned fields of at least one of the registers to produce a catenated result; and

wherein each of the multiple-bit data elements has an elemental width, and the data path has a data path width multiple times greater than the elemental width, to allow multiple-bit data elements used for the multiple instances of the arithmetic operation to be

transmitted in parallel from the register file to the execution unit, and wherein the execution unit is operable to receive, in parallel, multiple-bit data elements for the multiple instances of the arithmetic operation and execute the multiple instances of the arithmetic instruction to produce the catenated result (emphasis added)

Cray fails to teach or suggest at least three features recited in this claim: (1) a data path capable of parallel transmission of multiple data elements used for multiple instances of an arithmetic operation, (2) an execution unit capable of receiving multiple-bit data elements in parallel and executing multiple instances of an arithmetic operation to produce a catenated result, and (3) the execution unit being configured to execute a plurality of instruction streams from a plurality of threads. Cray's failure to disclose these three features is discussed in detail below.

A. Cray Fails to Disclose A Data Path Capable of Parallel Transmission of Multiple-bit Data Elements Used For Multiple Instances of An Arithmetic Operation

First, Cray fails to disclose a data path capable of parallel transmission of multiple-bit data elements used for multiple instances of an arithmetic operation. The Office Action points to Cray's vector floating point addition instruction ($V_j + V_k = V_i$) as supposedly disclosing the claimed "multiple instances of an arithmetic operation." See Office Action, p. 3, paragraph 2.¹ However, Cray's data path (Fig. 2, data path 23; Fig. 7, data paths 121, 123) is specifically designed to carry data elements for only one instance of the vector floating point add instruction at a time. For example, in one clock cycle, Cray's data path carries data element (V_0) for the first instance of the operation from the register to the vector functional unit. In the next clock cycle, the data path carries data element (V_1) for the second instance of the operation from the register to the vector functional unit, and so on. As Cray puts it:

¹ The Office Action states, at p. 3, paragraph 2: "...an arithmetic operation (addition; see col. 8, lines 28-35) to cause multiple instances of the operation to be performed, each instance of the arithmetic operation to be performed using a different one of the plurality of multiple-bit data elements (see col. 10, lines 19-50)" (referring to Cray's vector floating point addition instruction performed on vector elements V_i and V_j).

“In vector processing elements of one or more vector registers are successively transmitted as operands to a functional unit at a rate of one per clock period.” Cray, Abstract, lines 12-15 (emphasis added)

It telling that Cray specifically points out that in vector processing (the type of processing cited by the Office Action), data elements are successively transmitted over the data path to the vector functional unit at a rate of one per clock cycle, as opposed to being transmitted in parallel.

Indeed, the control circuitry disclosed by Cray for regulating the transfer of data elements over the data path, from the register to the vector functional unit, requires data elements for multiple instances of the vector operation to be transmitted one at a time, not in parallel. Fig. 7 of Cray illustrates this control circuitry and is reproduced below:

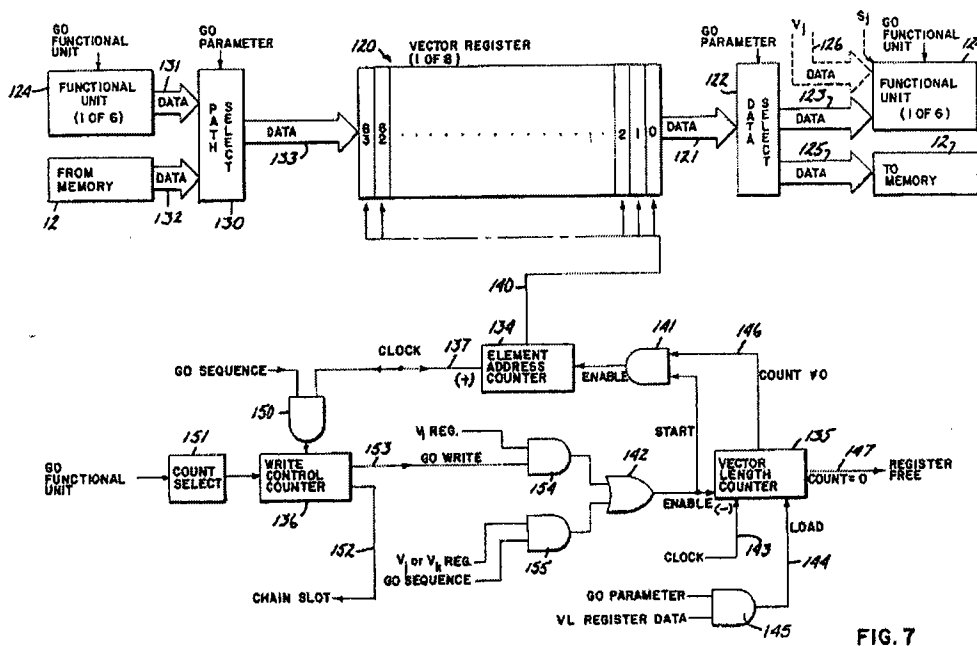


FIG. 7

As shown in this figure, an element address counter 134 selects data elements (V_0, V_1, V_2, \dots), one at a time, to be transmitted from the vector register 120 to function unit 124, over data paths 121 and 123. The element address counter 134 does so by generating an “element address” (140) that specifies which data element is to be selected for transfer from the

vector register. The element address counter 134 can select any data element from element V_0 up to element V_{63} (or alternatively, up to the length of the vector, as specified by vector length counter 135). However, only one data element is selected at a time. For example, in one clock cycle, the “element address” may be “0” (selecting data element V_0). In the next clock cycle, the “element address” may be “1” (selecting data element V_1), and so on. In this manner, the element address counter 134 successively selects, one at a time, data elements used for multiple instances of a vector instruction for transfer from the vector register 120 to functional unit 124.

Thus, Cray’s disclosure overwhelmingly indicates that its data path is incapable of parallel transmission of data elements used for multiple instances of an arithmetic operation from a register file to the execution unit, as required by claim 1. Cray itself expressly states that, for vector processing, elements are successively transmitted to the functional unit at a rate of one per clock period. Furthermore, Cray’s data path is regulated by control circuitry that only allows data elements to be successively transmitted (one at a time) for multiple instances of a vector operation. For at least this reason, claim 1 as amended overcomes the anticipation rejection based on Cray.

B. Cray Fails To Disclose An Execution Unit Capable of Receiving, in Parallel, Multiple-bit Data Elements Used for Multiple Instances of An Arithmetic Operation and Executing the Multiple Instances of the Arithmetic Operation to Produce a Catenated Result.

Second, Cray also fails to disclose an execution unit capable of receiving, in parallel, multiple-bit data elements used for multiple instances of an arithmetic operation and executing the multiple instances of the arithmetic operation to produce a catenated result. As discussed previously, the Office Action points to Cray’s vector floating point addition instruction ($V_j + V_k = V_i$) as supposedly disclosing the claimed “multiple instances of an arithmetic operation.” See Office Action, p. 3, paragraph 2. However, Cray’s vector functional unit (Fig. 2, vector functional unit; Fig. 7, functional unit 124) receives data elements for only one instance of the vector floating point add instruction at a time. Furthermore, Cray’s vector functional unit can only produce the result of one instance of the vector floating point add instruction at a time,

which is not a catenated result. The operation of Cray's vector functional unit is depicted in Fig. 3A, reproduced below:

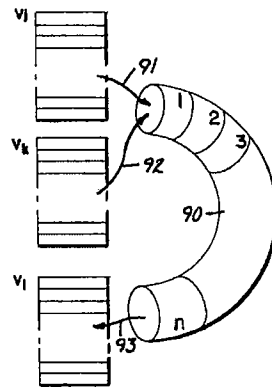


FIG. 3A

This figure illustrates the sequential nature in which the vector functional unit (represented by numeral 90) receives sets of data elements, one at a time, for a vector operation. Arrows 91 and 92 represent the vector functional unit's receipt of successive data elements from the V_j and V_k registers. For example, in clock cycle 1, the vector functional unit may receive data elements V_0 for a first instance of a vector floating point add instruction (via arrows 91 and 92). In clock cycle 2, the vector functional unit may receive data elements V_1 for a second instance of the vector floating point add instruction (again via arrows 91 and 92), and so on. Thus, Cray's vector functional unit is only capable of receiving data elements for just one instance of the vector floating point add instruction at a time. It cannot receive, in parallel, multiple-bit data elements used for multiple instances of a vector operation.

The figure above also illustrates the sequential nature in which the functional unit (represented by numeral 90) produces individual results, one at a time. After a delay of n clock cycles, the functional unit begins to successively output individual results of the vector operation. Arrow 93 represents the functional unit's output of these individual results. For example, in clock cycle n , the functional unit may output the result of the first instance of the vector floating point add instruction. In clock cycle $(n+1)$, the functional unit may output the result of the second instance of the vector floating point instruction. Clearly, the functional unit is designed to output the individual results of the vector instruction in a sequential fashion, not as

a catenated result. Thus, Cray's function unit cannot execute multiple instances of a vector operation to produce a catenated result.

As such, Cray fails to disclose an execution unit capable of receiving, in parallel, multiple-bit data elements used for multiple instances of an arithmetic operation and executing the multiple instances of the arithmetic operation to produce a catenated result, as required by claim 1. For this additional reason, claim 1 as amended overcomes the anticipation rejection based on Cray.

C. Cray Fails to Disclose an Execution Unit Configured to Execute a Plurality of Instruction Streams From a Plurality of Threads

Finally, Cray fails to disclose an execution unit that is configured to execute a plurality of instruction streams from a plurality of threads. Cray contains absolutely no mention of anything even remotely relating to such a limitation. However, the Office Action suggests that Cray inherently discloses the limitation, given the manner in which the limitation is recited in the original claim language.² Specifically, the Office Action states "any processor is *operable* to execute multiple streams." Office Action at p. 3, paragraph 2 (emphasis added). Applicants do not agree with this conclusory statement. However, in the interest of expediting prosecution of the present application, Applicants have amended the claim language to require an execution unit that is *configured* to execute a plurality of instruction stream from a plurality of threads.

Applicants respectfully assert that Cray fails to either expressly or inherently disclose the amended claim limitation of an execution unit that is *configured* to execute a plurality of instruction streams from a plurality of threads. Cray fails to expressly disclose this limitation, because Cray simply contains no teaching whatsoever that its vector functional unit is configured to execute a plurality of instruction streams from a plurality of threads.

Cray also fails to inherently disclose this limitation, as amended. The MPEP states that "in relying upon the theory of inherency, the examiner must provide a basis in fact

² As originally recited, the limitation reads "the execution unit *operable* to execute a plurality of instruction streams from the plurality of threads" (emphasis added).

and/or technical reasoning to reasonably support the determination that the allegedly inherent characteristic *necessarily* flows from the teaching of the applied prior art.” MPEP 2112(IV), page 2100-48 (emphasis in the original). In this case, it does not *necessarily* flow from the disclosure of Cray that its execution unit is configured to execute a plurality of instruction streams from a plurality of threads. In fact, Cray’s execution unit clearly is not configured in such a manner. Accordingly, Cray does not inherently disclose an execution unit that is configured in the manner required by the claim.

Thus, Cray fails to either expressly or inherently disclose an execution unit that is configured to execute a plurality of instruction streams from a plurality of threads. For this yet additional reason, claim 1 as amended overcomes the anticipation rejection based on Cray.

Claims 8, 13, 20, 27 and 28

Claims 8, 13, 20, 27 and 28 are rejected as purportedly anticipated by Cray under the same rationale as the rejection of claim 1. Claims 8, 13, 20, 27 and 28 have been amended and overcome the anticipation rejection based on Cray, for at least the reasons discussed previously with respect to claim 1.

Claims 6, 7, 12, 18, 19, 24, 25 and 26

Claims 6-7, 12, 18-19, 24 and 25-26 are rejected as purportedly anticipated by Cray. Claims 6 and 7 depend from claim 1. Claim 12 depends from claim 8. Claims 18 and 19 depend from claim 13. Claim 24 depends from claim 20. Claims 25 and 26 depend from claim 1. Claims 6, 7, 12, 18, 19, 24, 25 and 26 incorporate all of the limitations of their respective base claims. Accordingly, for at least the reasons stated previous with respect to claim 1, 8, 13 and 20, claims 6-7, 12, 18-19, 24 and 25-26 overcome the anticipation rejection based on Cray.

REJECTIONS UNDER 35 U.S.C. § 103

Claim 2

Claims 2 is rejected under 35 U.S.C. 103(a) as purportedly being unpatentable over Cray in view of Laudon et al. Claim 2 recites:

The processor of claim 1 wherein the execution unit comprises a pipeline having a plurality of stages and wherein the pipeline interleaves execution of instructions from the plurality of instruction streams

The Office Action concedes that Cray fails to disclose this limitation. *See* Office Action, p. 5, last paragraph. However, the Office Action contends that Laudon et al. teaches the limitation and proposes to combine such teachings from Laudon et al. with Cray.

Applicants submit that Cray cannot be combined with Laudon in the manner proposed by the Office Action, because Cray explicitly teaches away from such a combination. A prior art reference must be considered in its entirety, *i.e.*, as a whole, including portions that would lead away from the claimed invention. *W.L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 220 USPQ 303 (Fed. Cir. 1983), cert. denied, 469 U.S. 851 (1984). Here, Cray teaches away from the use of pipelining in an execution unit in the manner suggested by the Office Action. The Office Action likens the claimed “execution unit” to Cray’s vector functional unit when it is used to carries out a vector operation.³ However, when Cray’s vector functional unit is carrying out a vector operation, it can only do so in a “reserved” state that does not allow the vector functional unit to execute other instructions. As Cray states, at col. 8, lines 9-16:

A functional unit engaged in a vector operation remains busy during each clock period and may not participate in another operation. In this state, the functional unit is said to be reserved. Other instructions that require the same functional unit will not issue until the previous operation is complete. When the vector operation completes, the reservation is dropped and the functional unit is then available for another operation (emphasis added).

In such a “reserved” state, the vector functional unit is reserved for the execution of one and only one vector instruction. Before completion of the current vector instruction, other instructions requiring the same vector functional unit are not allowed to issue. Thus, Cray

³ As recited in claim 1 (limitations of which are incorporated in claim 2), the claimed “execution unit” must be “configured to execute a plurality of instruction streams...including a single instruction that specifies an arithmetic operation to cause multiple instances of the arithmetic operation to be performed, each instance of the arithmetic operation to be performed using a different one of the plurality of multiple-bit data elements in partitioned fields of at least one of the registers.” The Office Action contends that such a claimed “execution unit” is disclosed by Cray’s vector functional unit when it is used to carry out a vector operation -- *i.e.*, a vector floating point add instruction. *See* Office Action, p. 3, paragraph 2 (citing Cray at col. 8, lines 28-35 and col. 10, lines 19-50).

explicitly teaches away from the combination proposed by the Office Action – *i.e.*, to introduce interleaved execution of other instructions from other instruction streams (as supposedly taught by Laudon et al.) into Cray’s vector functional unit. Indeed, Cray’s reserved state does not even permit such other instructions to issue, let alone allow them to enter the vector functional unit for interleaved execution. Accordingly, Cray cannot be combined with Laudon et al. in the manner suggested by the Office Action. For at least this reason, claim 2 is patentable over Cray in view of Laudon et al.

Furthermore, claim 2 depends from claim 1 and therefore incorporates all of the limitations of claim 1. As discussed in previous sections, Cray fails to disclose at least three features of claim 1. Laudon et al. does not make up for these deficiencies. For at least this additional reason, claim 2 is patentable over Cray in view of Laudon et al.

Claim 3

Claims 3 is rejected under 35 U.S.C. 103(a) as purportedly being unpatentable over Cray in view of Laudon et al. Claim 3 recites:

The processor of claim 2 wherein the pipeline is operable to simultaneously contain states of execution of at least two instructions from different instruction streams

The Office Action concedes that Cray fails to disclose this limitation. *See* Office Action, p. 6, first paragraph. However, the Office Action contends that Laudon et al. teaches the limitation and proposes to combine such teachings from Laudon et al. with Cray.

Applicants submit that Cray cannot be combined with Laudon in the manner proposed by the Office Action. As discussed previously, Cray’s vector function unit operates in a “reserved” state when carrying out a vector operation. In such a “reserved” state, the vector functional unit is dedicated to the completion of one vector instruction. Before completion of the current vector instruction, no other instruction requiring the same vector functional unit is even allowed to issue. In this manner, Cray ensures that the pipeline does not simultaneously contain states of execution of more than one instruction. As such, Cray teaches away from operating a pipeline to simultaneously contain states of execution of at least two instructions from different

instruction streams (as supposedly taught by Laudon et al.). In other words, Cray cannot be combined with Laudon et al. in the manner suggested by the Office Action. For at least this reason, claim 3 is patentable over Cray in view of Laudon et al.

Furthermore, claim 3 depends from claim 2 and therefore incorporates all of the limitations of claim 2. As such, claim 3 is patentable over Cray in view of Laudon et al. for at least the reasons stated above with respect to claim 2, as well.

Claim 4

Claim 4 is rejected as purportedly being unpatentable over Cray in view of Laudon et al. Claim 4 depends from claim 2 and therefore incorporates all of the limitations of claim 2. As such, claim 4 is patentable over Cray in view of Laudon et al. for at least the reasons stated above with respect to claim 2.

Claim 5

Claim 5 is rejected as purportedly being unpatentable over Cray in view of Laudon et al. Claim 5 depends from claim 1 and therefore incorporates all of the limitations of claim 1. As discussed in previous sections, Cray fails to disclose at least three features of claim 1. Laudon et al. does not make up for these deficiencies. For at least this reason, claim 5 is patentable over Cray in view of Laudon et al.

Claims 9, 14 and 21

Claims 9, 14 and 21 are rejected as purportedly being unpatentable over Cray in view of Laudon et al., under the same rationale as the rejection of claim 2. For at least the reasons discussed above with respect to claim 2, claims 9, 14 and 21 are also patentable over Cray in view of Laudon et al.

Furthermore, claims 9, 14 and 21 depend from and incorporate all the limitations of claims 8, 13 and 20, respectively. As discussed previously, Cray fails to disclose various features recited in claims 8, 13 and 20. Laudon et al. does not make up for these deficiencies.

For at least this additional reason, claim 9, 14 and 21 are patentable over Cray in view of Laudon et al.

Claims 10, 15 and 22

Claims 10, 15 and 22 are rejected as purportedly being unpatentable over Cray in view of Laudon et al., under the same rationale as the rejection of claim 3. For at least the reasons discussed above with respect to claim 3, claims 10, 15 and 22 are also patentable over Cray in view of Laudon et al.

Furthermore, claims 10, 15 and 22 depend from and incorporate all of the limitations of claims 9, 14 and 21, respectively. As such, claims 10, 15 and 22 are patentable over Cray in view of Laudon et al. for at least the reasons stated above with respect to claims 9, 14 and 21, as well.

Claims 11, 16 and 23

Claims 11, 16 and 23 are rejected as purportedly being unpatentable over Cray in view of Laudon et al. Claims 11, 16 and 23 depend from and incorporate all of the limitations of claims 9, 14 and 21, respectively. As such, claims 11, 16 and 23 are patentable over Cray in view of Laudon et al. for at least the reasons stated above with respect to claims 9, 14 and 21.

Claims 12, 17, 18, 19, 24, 25 and 26

Claims 12, 17, 18, 19, 24, 25 and 26 are rejected as purportedly being unpatentable over Cray in view of Laudon et al. Claim 12 depends from claim 8. Claims 17-19 depend from claim 13. Claim 24 depends from claim 20. Claims 25-26 depend from claim 1. Claims 12, 17, 18, 19, 24, 25 and 26 incorporate all of the limitations of their respective base claims. Accordingly, for at least the reasons stated previous with respect to claim 8, 13 and 20, claims 12, 17, 18, 19, 24, 25 and 26 are patentable over Cray in view of Laudon et al.


CONCLUSION

In view of the foregoing, Applicants believe all claims now pending in this Application are in condition for allowance. The issuance of a formal Notice of Allowance at an early date is respectfully requested. If the Examiner believes a telephone conference would expedite prosecution of this application, please telephone the undersigned at 202-756-8363.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP


Kenneth L. Cage
Registration No. 26,151
Reg. 33,424

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 KLC:amz
Facsimile: 202.756.8087
Date: February 9, 2009

**Please recognize our Customer No. 20277
as our correspondence address.**